Remarks

Applicants thank the Examiner for examining the claims of the present application. By this amendment, claims 1, 5, 8, 13-14 and 16-19 are being amended; claims 3, 7, 9-12, 15, 20-23 and 25-33 are being canceled without prejudice; and claims 34-42 are being added. Support for these amendments can be found in the application, for example, at page 1, line 30 – page 2, line 16; page 8, lines 18-20; page 8, line 29 – page 9, line 4; page 15, line 19 – page 17, line 21; and page 21, lines 1-3. Upon entry of this amendment, claims 1, 4-5, 8, 13-14, 16-19, 24 and 34-42 will be pending. Applicants respectfully request reconsideration of the Examiner's rejections in view of the preceding amendments and the following remarks.

I. Examiner Interview

Applicants thank the Examiner for his time during a telephonic interview on August 21, 2009, during which Applicants' proposed amendments to claim 1 were discussed. Although no specific agreement was reached, the interview was helpful in advancing prosecution. Applicants believe that the claims presented herein should place the application in condition for allowance.

II. Claim Objections

The Examiner objects to claims 1, 19 and 33, contending that the "the phraseology written as 'target efficient testing of the program when executed' (cl. 1, 19, 33) and 'targeting testing' (cl. 1) amounts to idiomatic English constructs that cannot be unduly understood." (Office action, pg. 2.) Applicants respectfully disagree. However, solely to expedite prosecution, Applicants have amended claims 1, 19 and 33 to exclude the language "target efficient testing of the program when executed" and "targeting testing."

Accordingly, the Examiner's objections to claims 1, 19 and 33 should be withdrawn and such action is respectfully requested.

III. Claims 1, 4-5, 8, 13-14 and 16-18 are Not Unpatentable Under 35 U.S.C. § 112, ¶ 1

The Examiner rejects claims 1, 4-5, 8, 13-14 and 16-18 under 35 U.S.C. § 112, ¶ 1 for allegedly failing to comply with the written description requirement. (Office action, pgs. 2-4.) This rejection is traversed.

The Examiner contends that the language "reading a reflection of the executable ... program during a first execution of the program" in claim 1 "is not shown to have proper support or clear description in the Specifications." (Office action, pg. 3.) The Examiner also alleges that, in regards to the language "targeting testing ... **during** a second execution of the program" in claim 1, "[t]he *first* execution of a program followed by a *second* execution of that same program is not deemed as having explicit and reasonable support." (Office action, pgs. 3-4.) Applicants respectfully disagree. However, solely to expedite prosecution, Applicants have further amended independent claim 1 to remove the language "during a first execution" and "targeting testing during the second execution of the executable program."

Accordingly, the § 112, ¶ 1 rejection of claims 1, 4-5, 8, 13-14 and 16-18 should be withdrawn and such action is respectfully requested.

IV. Claims 1, 4-5, 8, 13-14, 16-19 and 24 are Patentable Over *Davidson* in View of *Java2SE*

The Action rejects claims 1, 4-5, 8, 13-14, 16-19 and 24 under 35 U.S.C § 103(a) as obvious over United States Patent No. 6,083,276 ("*Davidson*") in view of Class Introspector, Java 2 Platform Std. Ed. v.1.4.0, Sun Microsystems, copyright 1993-2002, pgs. 1-6 ("*Java2SE*"). (Office action, pgs. 4-14.) This rejection is traversed.

A. Independent Claim 1

Amended independent claim 1 recites a computer implemented method comprising, in relevant part:

> receiving domain configuration information comprising:
>
> ...
>
> one or more second mathematical or programmatic expressions related to configuring a second data domain corresponding to the second data structure element, the second mathematical or programmatic expressions operable on the one or more first data domains;
>
> ...
>
> producing the second data domain by evaluating the one or more second mathematical or programmatic expressions, the second data domain comprising one or more values of tuples generated from evaluating the one or more second mathematical or programmatic expressions; and
>
> wherein each data domain defines a set of values or tuples to be used for the corresponding data structure element during testing and validation of the computer program, and the second data domain contains at least one value or tuple not included in the one or more first data domains.

Applicants respectfully submit that *neither Davidson nor Java2SE (individually or in combination with each other) describes "second mathematical or programmatic expressions operable on the one or more first data domains ... producing the second data domain by evaluating the one or more second mathematical or programmatic expressions ... the second data domain contains at least one value or tuple not included in the one or more first data domains" as in amended independent claim 1.*

For example, *Davidson* describes "a method for creating and configuring a component-based application through text-based descriptive attribute grammar includ[ing] creating a parse tree from an application description file [ADF]" and "transforming the parse tree into a plurality of components corresponding to instances of classes in an application framework." (*Davidson*, Abstract.) The parse tree comprises elements, which "are associated with components 212 in the application framework" (*Davidson,* col. 7, lines 30-32). The elements "may contain attributes 310, which correspond to properties of a component 212. The attributes 310 and their values are parsed by the parser 116 and placed in an attribute list 312 within [an] element 306." (*Davidson*, col. 7, lines 36-40.)

Figure 4C of *Davidson* illustrates a flowchart for "mapping element attributes 310 to component properties 320." (*Davidson*, col. 5, lines 21-22.) The method comprises selecting "the next attribute 310 in the attribute list 312" (step 420), "checking ... the expected parameters of the write method 504 [of a component property 320] against the string value of the attribute 310" (step 430), converting the attribute value, if necessary (steps 432, 434, 438), and "executing 436 [the write method] with the attribute 310 value being passed as a parameter" (step 436). (*Davidson*, col. 25, line 11 - col. 26, line 3.) A parameter converter is used "for converting [attribute] string values into a variety of basic data types such as integers, doubles, dates, Booleans, and the likes" and also "may convert string-based component references into the corresponding component 212 references. For example, in one embodiment, a set of canonical object names may be used anywhere within a scoped usage path. These canonical object names include: [_SELF, _PARENT, etc]." (*Davidson*, col. 25, lines 33-58.)

However, *Davidson* fails to describe "second mathematical or programmatic expressions operable on the one or more first data domains ... producing the second data domain by evaluating the one or more second mathematical or programmatic expressions ... the second data domain contains at least one value or tuple not included in the one or more first data domains" as

in amended claim 1. *Davidson's* disclosure of "converting [attribute] string values into a variety of basic data types" simply does not teach converting the string value into another value. Similarly, *Davidson's* description of "convert[ing] string-based component references into corresponding component references" also fails to disclose converting attribute values. Thus, *Davidson* simply cannot produce a "data domain for the second data structure element comprising at least one data value or tuple not included in the at least one data domain for the one or more first data structure elements" as recited in claim 1. Accordingly, *Davidson* fails to describe "second mathematical or programmatic expressions operable on the one or more first data domains ... producing the second data domain by evaluating the one or more second mathematical or programmatic expressions ... the second data domain contains at least one value or tuple not included in the one or more first data domains" as in claim 1.

Java2SE similarly fails to teach this element. *Java2SE* discloses an "Introspector class" that "provides a standard way for tools to learn about the properties, events, and methods supported by a target Java Bean." (*Java2SE*, pg. 1.) The Introspector "analyze[s] [a] bean's class and superclasses" and "build[s] a BeanInfo object that comprehensively describes the target bean." However, *Java2SE* does not describe "second mathematical or programmatic expressions operable on the one or more first data domains ... producing the second data domain by evaluating the one or more second mathematical or programmatic expressions ... the second data domain contains at least one value or tuple not included in the one or more first data domains" as in claim 1. The Introspector only collects data for a target bean class, and does not "evaluat[e] mathematical or programmatic expressions" that are "operable on one of the one or more first data domains" as in claim 1. Thus, *Java2SE* cannot generate a "second data domain contain[ing] at least one value or tuple not included in the one or more first data domains" as recited in claim 1.

Accordingly, neither *Davidson* nor *Java2SE* (individually or in combination with each other) describes "second mathematical or programmatic expressions operable on the one or more first data domains ... producing the second data domain by evaluating the one or more second mathematical or programmatic expressions ... the second data domain contains at least one value or tuple not included in the one or more first data domains" as recited in amended independent claim 1.

For at least the above reasons, the 35 U.S.C. § 103(a) rejection of claim 1 should be withdrawn and such action is respectfully requested.

B. Dependent Claims 4-5, 8, 13-14 and 16-18

The Examiner also rejects dependent claims 4-5, 8, 13-14 and 16-18 as allegedly being obvious over *Davidson* in view of *Java2SE*. (Office action, pgs. 8-11.) Claims 4-5, 8, 13-14 and 16-18 depend from amended independent claim 1, and are allowable for at least the reasons discussed above with respect to amended independent claim 1. Furthermore, claims 4-5, 8, 13-14 and 16-18 are allowable because of the new and nonobvious combinations of features recited in each respective claim.

For at least these reasons, the 35 U.S.C. § 103(a) rejections of claims 4-5, 8, 13-14 and 16-18 should be withdrawn and such action is respectfully requested.

C. Independent Claim 19

Amended independent claim 19 recites, in relevant part:

> One or more computer-readable media storing computer-executable instructions for performing a method for producing sets of data values to be used for data structure elements of a computer program during testing and validation of the computer program the method comprising:
>
> ...
>
> receiving domain configuration information comprising:
>
> ...
>
> one or more second mathematical or programmatic expressions related to configuring a second data domain corresponding to the second data structure element, the second mathematical or programmatic expressions operating on the one or more first data domains;
>
> ...
>
> producing the second data domain by evaluating the one or more second mathematical or programmatic expressions, the second data domain comprising one or more values or tuples generated from evaluating the one or more second mathematical or programmatic expressions; and
>
> wherein each data domain defines a set of values or tuples for the corresponding data structure element to be used during testing and validation of the computer program, and the second data domain contains at least one value or tuple not included in the one or more first data domains.

Applicants respectfully *submit that neither Davidson nor Java2SE (individually or in combination with each other) describes "second mathematical or programmatic expressions operable on the one or more first data domains ... producing the second data domain by*

evaluating the one or more second mathematical or programmatic expressions ... the second data domain contains at least one value or tuple not included in the one or more first data domains" as in amended claim 19.

For example, *Davidson* describes "a method for creating and configuring a component-based application through text-based descriptive attribute grammar includ[ing] creating a parse tree from an application description file [ADF]" and "transforming the parse tree into a plurality of components corresponding to instances of classes in an application framework." (*Davidson*, Abstract.) The parse tree comprises elements, which "are associated with components 212 in the application framework" (*Davidson,* col. 7, lines 30-32). The elements "may contain attributes 310, which correspond to properties of a component 212. The attributes 310 and their values are parsed by the parser 116 and placed in an attribute list 312 within [an] element 306." (*Davidson*, col. 7, lines 36-40.)

Figure 4C of *Davidson* illustrates a flowchart for "mapping element attributes 310 to component properties 320." (*Davidson*, col. 5, lines 21-22.) The method comprises selecting "the next attribute 310 in the attribute list 312" (step 420), "checking ... the expected parameters of the write method 504 [of a component property 320] against the string value of the attribute 310" (step 430), converting the attribute value, if necessary (steps 432, 434, 438), and "executing 436 [the write method] with the attribute 310 value being passed as a parameter" (step 436). (*Davidson*, col. 25, line 11 - col. 26, line 3.) A parameter converter is used "for converting [attribute] string values into a variety of basic data types such as integers, doubles, dates, Booleans, and the likes" and also "may convert string-based component references into the corresponding component 212 references. For example, in one embodiment, a set of canonical object names may be used anywhere within a scoped usage path. These canonical object names include: [_SELF, _PARENT, etc]." (*Davidson*, col. 25, lines 33-58.)

However, *Davidson* fails to describe "second mathematical or programmatic expressions operable on the one or more first data domains ... producing the second data domain by evaluating the one or more second mathematical or programmatic expressions ... the second data domain contains at least one value or tuple not included in the one or more first data domains" as in amended claim 19. *Davidson's* disclosure of "converting [attribute] string values into a variety of basic data types" simply does not teach converting the string value into another value. Similarly, *Davidson's* description of "convert[ing] string-based component references into

corresponding component references" also fails to disclose converting attribute values. Thus, *Davidson* simply cannot produce a "data domain for the second data structure element comprising at least one data value or tuple not included in the at least one data domain for the one or more first data structure elements" as recited in claim 19. Accordingly, *Davidson* fails to describe "second mathematical or programmatic expressions operable on the one or more first data domains ... producing the second data domain by evaluating the one or more second mathematical or programmatic expressions ... the second data domain contains at least one value or tuple not included in the one or more first data domains" as in claim 19.

Java2SE similarly fails to teach this element. *Java2SE* discloses an "Introspector class" that "provides a standard way for tools to learn about the properties, events, and methods supported by a target Java Bean." (*Java2SE*, pg. 1.) The Introspector "analyze[s] [a] bean's class and superclasses" and "build[s] a BeanInfo object that comprehensively describes the target bean." However, *Java2SE* does not describe "second mathematical or programmatic expressions operable on the one or more first data domains ... producing the second data domain by evaluating the one or more second mathematical or programmatic expressions ... the second data domain contains at least one value or tuple not included in the one or more first data domains" as in claim 19. The Introspector only collects data for a target bean class, and does not "evaluat[e] mathematical or programmatic expressions" that are "operable on one of the one or more first data domains" as in claim 19. Thus, *Java2SE* cannot generate a "second data domain contain[ing] at least one value or tuple not included in the one or more first data domains" as recited in claim 19.

Accordingly, neither *Davidson* nor *Java2SE* (individually or in combination with each other) describes "second mathematical or programmatic expressions operable on the one or more first data domains ... producing the second data domain by evaluating the one or more second mathematical or programmatic expressions ... the second data domain contains at least one value or tuple not included in the one or more first data domains" as recited in amended independent claim 19.

For at least the above reasons, the 35 U.S.C. § 103(a) rejection of claim 19 should be withdrawn and such action is respectfully requested.

D. Dependent Claim 24

The Examiner also rejects dependent 24 as allegedly being obvious over *Davidson* in view of *Java2SE*. (Office action, pg. 14.) Claim 24 depends from amended independent claim 19, and is allowable for at least the reasons discussed above with respect to amended independent claim 19. Furthermore, claim 24 is allowable because of the new and nonobvious combinations of features recited in each respective claim.

For at least these reasons, the 35 U.S.C. § 103(a) rejection of claim 24 should be withdrawn and such action is respectfully requested.

Conclusion

For the reasons cited above, the application is believed to be in condition for allowance and such action is respectfully requested. If any issues remain in light of these remarks, and amendments, the Examiner is formally requested to contact the undersigned attorney to arrange for a telephonic interview. This request is being submitted under MPEP § 713.01, which indicates that an interview may be arranged in advance by a written request.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By _____
Stephen A. Wight
Registration No. 37,759